A Fixed-Parameter Algorithm for the Directed Feedback Vertex Set Problem

JIANER CHEN, YANG LIU, SONGJIAN LU
Department of Computer Science
Texas A&M University
College Station, TX 77843, USA
{chen|yangliu|sjlu}@cs.tamu.edu

BARRY O'SULLIVAN and IGOR RAZGON
Cork Constraint Computation Centre
Computer Science Department
University College Cork, Ireland
{b.osullivan|i.razgon}@cs.ucc.ie

Abstract

The (parameterized) feedback vertex set problem on directed graphs (i.e., the dfvs problem) is defined as follows: given a directed graph $G$ and a parameter $k$, either construct a feedback vertex set of at most $k$ vertices in $G$ or report that no such a set exists. It has been a well-known open problem in parameterized computation and complexity whether the dfvs problem is fixed-parameter tractable, i.e., whether the problem can be solved in time $f(k)n^{O(1)}$ for some function $f$. In this paper, we develop new algorithmic techniques that result in an algorithm with running time $4^k k! n^{O(1)}$ for the dfvs problem. Therefore, we resolve this open problem.

1 Introduction

Let $G$ be a directed graph. A feedback vertex set $F$ (briefly, FVS) for $G$ is a set of vertices in $G$ such that every directed cycle in $G$ contains at least one vertex in $F$, or equivalently, that the removal of $F$ from the graph $G$ leaves a directed acyclic graph (i.e., a DAG). The (parameterized) feedback vertex set problem on directed graphs (briefly, the dfvs problem) is defined as follows: given a directed graph $G$ and a parameter $k$, either construct an FVS of at most $k$ vertices for $G$ or report that no such set exists.

The dfvs problem is a classic NP-complete problem that appeared in the first list of NP-complete problems in Karp's seminal paper [18], and has a variety of applications in areas such as operating systems [24], database systems [13], and circuit testing [20]. In particular, the dfvs problem has played an essential role in the study of deadlock recovery in database systems and in operating systems [24, 13]. In such a system, the status of system resource allocations can be represented as a directed graph $G$ (i.e., the system resource-allocation graph), and a directed cycle in $G$ represents a deadlock in the system. Therefore, in order to recover from deadlocks, we need to abort a set of processes in the system, i.e., to remove a set of vertices in the graph.
so that all directed cycles in $G$ are broken. Equivalently, we need to find an FVS in the graph $G$. In practice, one may expect and desire that the number of vertices removed from the graph $G$, which is the number of processes to be aborted in the system, be small. This motivates the study of parameterized algorithms for the dfvs problem that find an FVS of $k$ vertices in a directed graph of $n$ vertices and run in time $f(k)n^{O(1)}$ for a fixed function $f$; thus, the algorithms become practically efficient when the value $k$ is small.

This work has been part of a systematic study of the theory of fixed-parameter tractability [11], which has received considerable attention in recent years. A problem $Q$ is a parameterized problem if each instance of $Q$ contains a specific integral parameter $k$. A parameterized problem is fixed-parameter tractable if it can be solved in time $f(k)n^c$ for a function $f(k)$ and a constant $c$, where the function $f(k)$ is independent of the instance size $n$. A large number of NP-hard parameterized problems, such as the VERTEX COVER problem [6] and the ML TYPE-CHECKING problem [21], have been shown to be fixed-parameter tractable. On the other hand, strong evidence has been given that another group of well-known parameterized problems, including the INDEPENDENT SET problem and the DOMINATING SET problem, are not fixed-parameter tractable [11]. The study of fixed-parameter tractability of parameterized problems has become increasingly interesting, for both theoretical research and practical computation.

The fixed-parameter tractability of the dfvs problem was posted as an open problem in the very first papers on the study of fixed-parameter tractability [9, 10]. After numerous significant efforts, however, the problem still remained open. In the past fifteen years, the problem has been constantly and explicitly posted as an open problem in a large number of publications in the literature (see [17] for a recent survey on this study). The problem has become a well-known and outstanding open problem in parameterized computation and complexity.

In this paper, we develop new algorithmic techniques that lead to the conclusion that the dfvs problem is fixed-parameter tractable, and thus resolve the above open problem in parameterized computation and complexity. We first show that the dfvs problem can be reduced in time $f(k)n^{O(1)}$ for some function $f$ to a special version of the multi-cut problem, which will be called the skew separator problem. We then develop an algorithm that shows the fixed-parameter tractability of the skew separator problem. The combination of these two results gives an algorithm with running time $4^k k! n^{O(1)}$ for the dfvs problem, which proves its fixed-parameter tractability.

The relationship between the dfvs problem and multi-cut problems has been studied in the research of approximation algorithms for the FEEDBACK VERTEX SET problem [12, 19]. However, our problem formulations and the corresponding techniques are significantly different from those studied in the approximation algorithms. In particular, our formulations and techniques seem especially suitable for developing faster and more effective exact algorithms (of exponential-time) for NP-hard multi-cut problems. First of all, instead of seeking a multi-cut that separates a given set of terminal vertices, as formulated in most multi-cut problems, our problem is more general: we wish to construct a multi-cut that separates a collection of terminal vertex-subsets. This more general version of the multi-cut problem enables us to effectively reduce the search space size when we are searching for an optimal solution of a given problem instance. Secondly, unlike most multi-cut problems whose solutions are multi-cuts that are in general
symmetric to the given terminal vertices, the multi-cuts for the skew separator problem are asymmetric to the terminal vertex-subsets. Thirdly, we develop an (exponential-time) reduction that effectively reduces the problem of multi-cuts for multiple terminal vertex-subsets to the problem of minimum cuts from a single source vertex to a single sink vertex. Note that the latter is solvable in polynomial time via algorithms for the maximum flow problem. Such an exponential-time reduction is obviously very different from the polynomial-time processes used in the development of polynomial-time approximation algorithms. Finally, unlike most parameterized algorithms that are focused on effectively decreasing the sole parameter value $k$, our algorithm for the skew separator problem adds another dimension of bounds in terms of the size of a minimum cut between two properly chosen terminal vertex-subsets. This dimension of bounds has become crucial in our development of the algorithm for the skew separator problem because it effectively bounds the number of branches in which the parameter value $k$ is not decreased.

Before we move to the technical discussion of our algorithms, we remark that the feedback vertex set problem on undirected graphs (briefly, the ufvs problem) has also been an interesting and active research topic in parameterized computation and complexity. Since the first fixed-parameter tractable algorithm for the ufvs problem was published almost twenty years ago [2], there has been an impressive list of improved algorithms for the problem. Currently the best algorithm for the ufvs problem runs in time $O(5^k kn^2)$ [5]. The feedback vertex set problem on directed graphs (i.e., the dfvs problem) seems very different from the problem on undirected graphs (i.e., the ufvs problem). This fact has also been reflected in the study of approximation algorithms for the problems. The feedback vertex set problem on undirected graphs is polynomial-time approximable with a ratio 2. This holds true even for weighted graphs [1]. On the other hand, it still remains open whether the feedback vertex set problem on directed graphs has a constant-ratio polynomial-time approximation algorithm. The current best polynomial-time approximation algorithm for the problem on directed graphs has a ratio $O(\log \tau \log \log \tau)$, where $\tau$ is the size of a minimum FVS for the input graph [12].

2 Preliminaries

Let $G = (V, E)$ be a directed graph and let $e = [u, v]$ be a (directed) edge in $G$. We say that the edge $e$ goes out from the vertex $u$ and comes into the vertex $v$. The edge $e$ is called an outgoing edge of the vertex $u$, and an incoming edge of the vertex $v$. These concepts can be extended from single vertices to general vertex sets. Thus, for two vertex sets $S_1$ and $S_2$, we can say that an edge goes out from $S_1$ and comes into $S_2$ if the edge goes out from a vertex in $S_1$ and comes into a vertex in $S_2$. Moreover, we say that an edge goes out from $S_1$ if the edge goes out from a vertex in $S_1$ and comes into a vertex not in $S_1$, and that an edge comes into $S_2$ if the edge goes out from a vertex not in $S_2$ and comes into a vertex in $S_2$.

A path $P$ from a vertex $v_1$ to a vertex $v_h$ in the graph $G$ is a sequence $\{v_1, v_2, \ldots, v_h\}$ of vertices in $G$ such that $[v_i, v_{i+1}]$ is an edge in $G$ for all $1 \leq i \leq h - 1$. The path $P$ is simple if no vertex is repeated in $P$. The path $P$ is a cycle if $v_1 = v_h$, and the cycle is simple if no other vertices are repeated. We say that a path is from a vertex set $S_1$ to a vertex set $S_2$ if the path
is from a vertex in $S_1$ to a vertex in $S_2$. The graph $G$ is a DAG (i.e., directed acyclic graph) if it contains no cycles.

For a vertex subset $V' \subseteq V$ in the directed graph $G = (V, E)$, we denote by $G[V']$ the subgraph of $G$ that is induced by the vertex subset $V'$. Without any ambiguity, we will denote by $G - V'$ the induced subgraph $G[V - V']$, and by $G - w$, where $w$ is a vertex in $G$, the induced subgraph $G[V - \{w\}]$.

A vertex subset $F$ in the directed graph $G$ is a feedback vertex set (FVS) if the graph $G - F$ is a DAG. Since a vertex $v$ with a self-loop (i.e., an edge that both goes out from and comes into $v$) must be included in every FVS for the graph $G$, we will assume, without loss of generality, that the graphs in our discussion have no self-loops.

**Definition**  Let $[S_1, \ldots, S_l]$ and $[T_1, \ldots, T_l]$ be two collections of $l$ vertex subsets in a directed graph $G = (V, E)$. A skew separator $X$ for the pair $([S_1, \ldots, S_l], [T_1, \ldots, T_l])$ is a vertex subset in $V - \bigcup_{i=1}^{l}(S_i \cup T_i)$ such that for any pair of indices $i$ and $j$ satisfying $i \geq j \geq j \geq 1$, there is no path from $S_i$ to $T_j$ in the graph $G - X$.

The subsets $S_1$, ..., $S_l$ will be called the source sets and the subsets $T_1$, ..., $T_l$ will be called the sink sets. A vertex is a non-terminal vertex if it is not in $\bigcup_{i=1}^{l}(S_i \cup T_i)$. Note that by definition, all vertices in a skew separator must be non-terminal vertices. Moreover, a skew separator $X$ is asymmetric to the source sets and the sink sets: a path from $S_i$ to $T_j$ with $i < j$ may exist in the graph $G - X$.

When there is only one source set $S_1$ and one sink set $T_1$, a skew separator for the pair $([S_1], [T_1])$ becomes a regular cut for $S_1$ and $T_1$, i.e., a vertex set whose removal leaves a graph in which there is no path from the set $S_1$ to the set $T_1$. Therefore, a skew separator for the pair $([S_1], [T_1])$ is also called a cut from $S_1$ to $T_1$. A cut from $S_1$ to $T_1$ is a min-cut (i.e., a minimum cut) if it has the smallest cardinality over all cuts from $S_1$ to $T_1$.

The following lemma can be easily derived based on standard maximum flow techniques [23]. Thus, we omit its proof.

**Lemma 2.1** There is an $O(kn^2)$ time algorithm that for two given vertex subsets $S$ and $T$ in a directed graph $G$ of $n$ vertices, and a parameter $k$, either constructs a min-cut from $S$ to $T$ whose size is bounded by $k$, or reports that the min-cut from $S$ to $T$ has a size larger than $k$.

The algorithm for the DFVS problem is obtained through careful development of algorithms for a series of problems. In the following, we give the formal definitions of these problems.

**SKEW SEPARATOR:** given $(G, [S_1, \ldots, S_l], [T_1, \ldots, T_l], k)$, where $G$ is a directed graph, $[S_1, \ldots, S_l]$ is a collection of $l$ source sets and $[T_1, \ldots, T_l]$ is a collection of $l$ sink sets in $G$, and a parameter $k$, such that

1. all sets $S_1$, ..., $S_l$, $T_1$, ..., $T_l$ are pairwise disjoint;
2. for each $i$, $1 \leq i \leq l - 1$, there is no edge coming into the source set $S_i$; and
3. for each $j$, $1 \leq j \leq l$, there is no edge going out from the sink set $T_j$,
either construct a skew separator of at most \(k\) vertices for the pair \([S_1, \ldots, S_l], [T_1, \ldots, T_l]\),
or report that no such separator exists.

Note that in an instance of the **skew separator** problem, condition (2) on source sets and condition (3) on sink sets are not completely symmetric. Although the first \(l-1\) source sets are not allowed to have incoming edges, the last source set \(S_l\) is allowed to have incoming edges. On the other hand, all sink sets are not allowed to have outgoing edges.

We remark that conditions (1)-(3) in the definition of the **skew separator** problem (plus the restriction that the skew separator can consist of only non-terminal vertices) may be relaxed, and our techniques for the problem may still be applicable. However, the above formulation of the problem will make our discussion simpler, and will also be sufficient for our solution to the **dfvs** problem, which is the focus of the current paper. We leave the investigation of the separator problems of more general forms to later research.

Let \(G = (V, E)\) be a directed graph, and let \((D_1, D_2)\) be a bi-partition of the vertex set \(V\) of \(G\), i.e., \(D_1 \cup D_2 = V\) and \(D_1 \cap D_2 = \emptyset\). The bi-partition \((D_1, D_2)\) is a **DAG-bipartition** for the graph \(G\) if both induced subgraphs \(G[D_1]\) and \(G[D_2]\) are DAGs. A vertex subset \(F\) in the graph \(G\) is a **D\(1\)-FVS** if \(F\) is an FVS for \(G\) and \(F \subseteq D_1\).

**DAG-bipartition FVS**: given \((G, D_1, D_2, k)\), where \(G\) is a directed graph, \((D_1, D_2)\) is a DAG-bipartition for \(G\), and \(k\) is the parameter, either construct a \(D_1\)-FVS of size bounded by \(k\) for the graph \(G\), or report that no such \(D_1\)-FVS exists.

We will be also interested in a special version of the **feedback vertex set** problem.

**DFVS REDUCTION**: given a triple \((G, F, k)\), where \(G\) is a directed graph and \(F\) is an FVS of size \(k+1\) for \(G\), either construct an FVS of size bounded by \(k\) for \(G\), or report that no such FVS exists.

Finally, our central problem in this paper is as follows.

**DFVS**: given a pair \((G, k)\), where \(G\) is a directed graph and \(k\) is the parameter, either construct an FVS of size bounded by \(k\) for \(G\), or report that no such FVS exists.

### 3 Solving the **skew separator** problem

In this section, we study the complexity of the **skew separator** problem.

Let \((G, [S_1, \ldots, S_l], [T_1, \ldots, T_l], k)\) be an instance of the **skew separator** problem. Define \(T_{all} = \bigcup_{1 \leq i \leq l} T_i\). There are a few cases in which we can directly reduce the instance size:

**Rule R1**. There is no path from \(S_i\) to \(T_{all}\), i.e., the size of a min-cut from \(S_i\) to \(T_{all}\) is 0: then we only need to find a skew separator of size \(k\) that separates \(S_i\) from \(T_j\) for all indices \(i\) and \(j\) satisfying \(l-1 \geq i \geq j \geq 1\), i.e., we can work instead on the instance \((G, [S_1, \ldots, S_{l-1}], [T_1, \ldots, T_{l-1}], k)\). Note that in this case, by definition, if \(l = 1\), then the solution to the instance \((G, [S_1], [T_1], k)\) is simply the empty set \(\emptyset\).
**Rule R2.** There is an edge from \( S_l \) to \( T_{all} \): then there is no way to even separate \( S_l \) from \( T_{all} \) – we can simply stop and claim that the given instance is a “No” instance;

**Rule R3.** There exists a non-terminal vertex \( w \), an edge from \( S_l \) to \( w \), and an edge from \( w \) to \( T_{all} \): then the vertex \( w \) must be included in the skew separator in order to separate \( S_l \) and \( T_{all} \) – we can simply work on the instance \((G - w, [S_1, \ldots, S_l], [T_1, \ldots, T_l], k - 1)\) and recursively find a skew separator of size \( k - 1 \).

Note that in Rules R1 and R3, the reduced instances \((G, [S_1, \ldots, S_{l-1}], [T_1, \ldots, T_{l-1}], k)\) and \((G - w, [S_1, \ldots, S_l], [T_1, \ldots, T_l], k - 1)\) are still valid instances of the skew separator problem.

In the following discussion, assume that for the input instance \((G, [S_1, \ldots, S_l], [T_1, \ldots, T_l], k)\), none of the rules above is applicable. In particular, since Rule R1 is not applicable, a min-cut from \( S_l \) to \( T_{all} \) has size larger than 0. Because Rules R1-R3 are not applicable, there must be a non-terminal vertex \( u_0 \) such that (1) there is an edge from \( S_l \) to \( u_0 \); and (2) there is no edge from \( u_0 \) to \( T_{all} \). Such a vertex \( u_0 \) will be called an \( S_l \)-extended vertex. Fix an \( S_l \)-extended vertex \( u_0 \), let \( S'_l = S_l \cup \{u_0\} \).

We start with the following simple but important lemma. The proof of this lemma is straightforward. Thus, we leave it to the reader.

**Lemma 3.1** Let \( X \) be a subset of vertices in the graph \( G \) that does not contain the \( S_l \)-extended vertex \( u_0 \). Then \( X \) is a skew separator for the pair \(([S_1, \ldots, S_l], [T_1, \ldots, T_l])\) if and only if \( X \) is a skew separator for the pair \(([S_1, \ldots, S_{l-1}, S'_l], [T_1, \ldots, T_{l-1}, T_l])\).

Lemma 3.1 also directly implies the following two useful corollaries.

**Corollary 3.2** A skew separator for the pair \(([S_1, \ldots, S_{l-1}, S'_l], [T_1, \ldots, T_{l-1}, T_l])\) is also a skew separator for the pair \(([S_1, \ldots, S_l], [T_1, \ldots, T_l])\).

**Corollary 3.3** The size of a min-cut from \( S'_l \) to \( T_{all} \) in the graph \( G \) is at least as large as the size of a min-cut from \( S_l \) to \( T_{all} \) in \( G \).

Now we are ready for our main theorem in this section.

**Theorem 3.4** If the size of a min-cut from \( S_l \) to \( T_{all} \) is equal to the size of a min-cut from \( S'_l \) to \( T_{all} \), then the pair \(([S_1, \ldots, S_l], [T_1, \ldots, T_l])\) has a skew separator of size bounded by \( k \) if and only if the pair \(([S_1, \ldots, S_{l-1}, S'_l], [T_1, \ldots, T_{l-1}, T_l])\) has a skew separator of size bounded by \( k \).

**Proof.** \( \Leftarrow \): Suppose that the pair \(([S_1, \ldots, S_{l-1}, S'_l], [T_1, \ldots, T_{l-1}, T_l])\) has a skew separator \( X' \) of size bounded by \( k \). By Corollary 3.2, \( X' \) is also a skew separator for \(([S_1, \ldots, S_l], [T_1, \ldots, T_l])\). In consequence, \(([S_1, \ldots, S_l], [T_1, \ldots, T_l])\) has a skew separator of size bounded by \( k \).

\( \Rightarrow \): Suppose that the pair \(([S_1, \ldots, S_l], [T_1, \ldots, T_l])\) has a skew separator \( X \) of size bounded by \( k \). If the skew separator \( X \) does not contain the \( S_l \)-extended vertex \( u_0 \), then by Lemma 3.1, \( X \) is also a skew separator of size bounded by \( k \) for the pair \(([S_1, \ldots, S_{l-1}, S'_l], [T_1, \ldots, T_{l-1}, T_l])\), and the theorem is proved. Therefore, we can assume that the set \( X \) contains the \( S_l \)-extended vertex \( u_0 \). We will define another set \( X' \) that does not contain \( u_0 \). We will show that \(|X'| \leq |X|\).
and that $X'$ is a skew separator for $([S_1, \ldots, S_l], [T_1, \ldots, T_l])$. Then the theorem will immediately follow.

Let $Y$ be a min-cut from $S_l'$ to $T_{all}$. Then $Y$ does not contain the $S_l$-extended vertex $u_0$. Moreover, since there is no edge coming into $S_l$ from outside of $S_l$ for all $i \leq l - 1$, the set $Y$ does not contain any vertex in $\bigcup_{i=1}^{l-1} S_i$. In consequence, the set $Y$ consists of only non-terminal vertices. By Corollary 3.2, $Y$ is also a cut from $S_l$ to $T_{all}$. Moreover, by the assumption of the theorem that the size of a min-cut from $S_l$ to $T_{all}$ is equal to the size of a min-cut from $S_l'$ to $T_{all}$, $Y$ is actually also a min-cut from $S_l$ to $T_{all}$. Let $R_X(S_l)$ be the set of vertices $v$ such that $v \in S_l$ or there is a path from $S_l$ to $v$ in the subgraph $G - Y$. In particular, $u_0 \in R_X(S_l)$ because $Y$ does not contain $u_0$ and there is an edge from $S_l$ to $u_0$.

We introduce a number of sets as follows.

\[
Z = X \cap Y; \\
X_{in} = X \cap R_X(S_l); \\
X_{out} = X - (X_{in} \cup Z).
\]

That is, the skew separator $X$ for $([S_1, \ldots, S_l], [T_1, \ldots, T_l])$ is decomposed into three disjoint subsets $Z$, $X_{in}$, and $X_{out}$ (note that by definitions, $R_X(S_l)$ and $Y$ do not intersect).

Let $Y_T$ be the set of vertices $v$ in the min-cut $Y$ such that there is a path from $v$ to $T_{all}$ in the subgraph $G - X$. By definition, we have $Y_T \cap Z = \emptyset$. Let

\[
Y_S = Y - (Y_T \cup Z).
\]

Thus, the min-cut $Y$ from $S_l$ to $T_{all}$ is decomposed into three disjoint subsets $Z$, $Y_T$, and $Y_S$.

Figure 1 gives an intuitive illustration of the sets $Z$, $X_{in}$, $X_{out}$, $Y_T$, $Y_S$, and $R_X(S_l)$.

![Figure 1: Sets in the proof of Theorem 3.4.](image-url)
We first show that the set \( Y' = Y_S \cup Z \cup X_{in} \) is also a cut from \( S_l \) to \( T_{all} \). If by contradiction \( Y' \) is not a cut from \( S_l \) to \( T_{all} \), then there is a path \( P_1 \) from \( S_l \) to \( T_{all} \) in the subgraph \( G - Y' \). The path \( P_1 \) must contain vertices in the set \( Y \) since \( Y \) is a cut from \( S_l \) to \( T_{all} \). Let \( w \) be the first vertex on the path \( P_1 \) that is in \( Y \) when we traverse from \( S_l \) to \( T_{all} \) along the path \( P_1 \). Then \( w \) must be in \( Y_T \) since \( Y' \) contains both \( Y_S \) and \( Z \). Now the partial path \( P_1' \) of \( P_1 \) from \( S_l \) to \( w \) (not including \( w \)) must be entirely contained in \( R_Y(S_l) \) (note that the path \( P_1 \) does not intersect \( Y_S \cup Z \)). Moreover, the path \( P_1' \) contains neither vertices in \( X_{in} \cup Z \) (by the definition of the set \( Y' \)) nor vertices in \( X_{out} \) (since the sets \( X_{out} \) and \( R_Y(S_l) \) are disjoint). In summary, the subpath \( P_1' \) from \( S_l \) to \( w \) contains no vertex in the set \( X \). Moreover, by the definition of the set \( Y_T \), and \( w \in Y_T \), there is a path \( P_1'' \) from \( w \) to \( T_{all} \) in the subgraph \( G - X \). Now the concatenation of the paths \( P_1' \) and \( P_1'' \) would result in a path from \( S_l \) to \( T_{all} \) in the graph \( G - X \), contradicting the fact that \( X \) is a skew separator for the pair \( ([S_l, \ldots, S_l], [T_1, \ldots, T_i]) \). This contradiction shows that the set \( Y' \) must be a cut from \( S_l \) to \( T_{all} \).

Since \( Y \) is a min-cut from \( S_l \) to \( T_{all} \), we have \( |Y| \leq |Y'| \). By definition, \( Y = Y_S \cup Z \cup Y_T \) and \( Y' = Y_S \cup Z \cup X_{in} \). Also note that \( Y_S \), \( Z \), and \( Y_T \) are pairwise disjoint, and that \( Y_S \), \( Z \), and \( X_{in} \) are also pairwise disjoint. Therefore, we must have \( |Y_T| \leq |X_{in}| \).

Consider the set \( X' = X_{out} \cup Z \cup Y_T \). The set \( X' \) has the following properties: (1) \( X' \) consists of only non-terminal vertices (because both \( X \) and \( Y \) consist of only non-terminal vertices); (2) \( |X'| \leq |X| \) (because \( |Y_T| \leq |X_{in}| \)), so the size of \( X' \) is bounded by \( k \); and (3) the set \( X' \) does not contain the \( S_l \)-extended vertex \( u_0 \) (this is because \( u_0 \) is in \( X_{in} \) and \( Y \) does not contain \( u_0 \)). Therefore, if we can prove that \( X' \) is a skew separator for the pair \( ([S_1, \ldots, S_l], [T_1, \ldots, T_i]) \), then by Lemma 3.1, \( X' \) is also a skew separator of size bounded by \( k \) for the pair \( ([S_1, \ldots, S_{l-1}, S_l'], [T_1, \ldots, T_{i-1}, T_i]) \). This will complete the proof of the theorem.

Therefore, what remains is to prove that the set \( X' = X_{out} \cup Z \cup Y_T \) is a skew separator for the pair \( ([S_1, \ldots, S_l], [T_1, \ldots, T_i]) \). Let \( R_Y(T_{all}) \) be the set of vertices \( v \) such that either \( v \in T_{all} \) or there is a path from \( v \) to \( T_{all} \) in the subgraph \( G - Y \).

Suppose by contradiction that \( X' \) is not a skew separator for \(([S_1, \ldots, S_l], [T_1, \ldots, T_i]) \). Then there is a path \( P_2 \) in the subgraph \( G - X' \) from \( S_l \) to \( T_j \) for some \( i \geq j \). The path \( P_2 \) has the following properties:

1. The path \( P_2 \) must contain a vertex in \( R_Y(S_l) \): since \( X \) is a skew separator for the pair \( ([S_1, \ldots, S_l], [T_1, \ldots, T_i]) \), the path \( P_2 \) from \( S_l \) to \( T_j \) with \( i \geq j \) must contain at least one vertex \( w_1 \) in \( X = X_{in} \cup Z \cup X_{out} \). Now since the path \( P_2 \) is in the subgraph \( G - X' \), where \( X' = X_{out} \cup Z \cup Y_T \), the vertex \( w_1 \) must be in \( X_{in} \), which is a subset of \( R_Y(S_l) \);

2. The path \( P_2 \) must contain a vertex in \( Y_S \): by Property 1, \( P_2 \) contains a vertex \( w_1 \) in \( R_Y(S_l) \). From the vertex \( w_1 \) to \( T_{all} \) along the path \( P_2 \), there must be a vertex \( w_2 \) in \( Y = Y_S \cup Z \cup Y_T \) since \( Y \) is a cut from \( S_l \) to \( T_{all} \) while \( w_1 \) is reachable from \( S_l \) in the subgraph \( G - Y \). Now since \( X' = X_{out} \cup Z \cup Y_T \), and the path \( P_2 \) is in the subgraph \( G - X' \), the vertex \( w_2 \) on the path \( P_2 \) must be in the set \( Y_S \);

3. The path \( P_2 \) must end at a vertex in \( R_Y(T_{all}) \): this is simply because \( P_2 \) is ended in \( T_{all} \). Note that by definition, no vertex in \( Y_S \) can be in \( R_Y(T_{all}) \).
By Properties 2-3, the path $P_2$ contains a vertex not in $R_Y(T_{all})$ and ends at a vertex in $R_Y(T_{all})$. Thus, there must be an internal vertex $w$ in the path such that $w$ is not in $R_Y(T_{all})$ but all vertices after $w$ along the path $P_2$ (from $S_i$ to $T_j$) are in $R_Y(T_{all})$. Note that no vertex $w'$ after the vertex $w$ along the path $P_2$ can be in the set $X$: $w'$ in $X$ would imply $w'$ in $X_m$ (since $P_2$ is a path in the subgraph $G - X'$), which would imply that there is another vertex after $w'$ that is in $Y$ thus is not in $R_Y(T_{all})$. Moreover, the vertex $w$ must be in the set $Y$ (otherwise, $w$ would be in $R_Y(T_{all})$). Since $P_2$ is a path in $G - X'$ and $X' = X_{out} \cup Z \cup Y_T$, the vertex $w$ must be in the set $Y_S$. However, this derives a contradiction: the subpath of $P_2$ from $w$ to $T_{all}$ shows that the vertex $w$ should belong to the set $Y_T$ (note that all vertices after $w$ on the path are not in $X$), and the sets $Y_S$ and $Y_T$ are disjoint. This contradiction proves that the set $X'$ must be a skew separator for the pair $([S_1, \ldots, S_l], [T_1, \ldots, T_l])$. Since the size of the set $X'$ is bounded by $k$ and $X'$ does not contain the $S_i$-extended vertex $u_0$, by Lemma 3.1, the set $X'$ is also a skew separator for the pair $([S_1, \ldots, S_{l-1}, S'_l], [T_1, \ldots, T_{l-1}, T_l])$, and the size of $X'$ is bounded by $k$.

This completes the proof of the theorem. □

Theorem 3.4 enables us to develop a parameterized algorithm for the skew separator problem. The algorithm is presented in Figure 2.

Figure 2: An algorithm for the skew separator problem.

**Algorithm SMC**($G, [S_1, \ldots, S_l], [T_1, \ldots, T_l], k$)

input: an instance ($G, [S_1, \ldots, S_l], [T_1, \ldots, T_l], k$) of the skew separator problem.

output: a skew separator of size bounded by $k$ for the pair $([S_1, \ldots, S_l], [T_1, \ldots, T_l])$, or report “No” (i.e., no such separator exists).

1. if $l = 1$ then solve the problem in time $O(kn^2)$;
2. if Rule R2 applies or $k < 0$ then return “No”;
3. if Rule R1 applies then return SMC($G, [S_1, \ldots, S_{l-1}], [T_1, \ldots, T_{l-1}], k$);
4. if Rule R3 applies on a vertex $w$ then return $\{w\} \cup$ SMC($G - w, [S_1, \ldots, S_l], [T_1, \ldots, T_l], k - 1$); \(^5\)
5. pick an $S_i$-extended vertex $u_0$; let $S'_i = S_i \cup \{u_0\}$;
6. let $m$ be the size of a min-cut from $S_i$ to $T_{all} = \bigcup_{i=1}^l T_i$;
7. if $m > k$ then return “No”;
8. let $m'$ be the size of a min-cut from $S'_i$ to $T_{all}$;
9. if $(m = m')$
10. then return SMC($G, [S_1, \ldots, S_{l-1}, S'_l], [T_1, \ldots, T_{l-1}, T_l], k$);
11. else $X = \{u_0\} \cup$ SMC($G - u_0, [S_1, \ldots, S_l], [T_1, \ldots, T_l], k - 1$);
12. if $X \neq “No”$ then return $X$;
13. else return SMC($G, [S_1, \ldots, S_{l-1}, S'_l], [T_1, \ldots, T_{l-1}, T_l], k$).

\(^5\) To simplify our description, we assume that a “No” plus anything gives a “No”.

**Theorem 3.5** The algorithm SMC($G, [S_1, \ldots, S_l], [T_1, \ldots, T_l], k$) solves the skew separator problem in $O(4^k kn^3)$ time, where $n$ is the number of vertices in the input graph $G$.

**Proof.** We first prove the correctness of the algorithm. Let ($G, [S_1, \ldots, S_l], [T_1, \ldots, T_l], k$)
be an input to the algorithm, which is an instance of the skew separator problem, where
\( G = (V, E) \) is a directed graph, \([S_1, \ldots, S_l]\) and \([T_1, \ldots, T_l]\) are the source sets and the sink sets, respectively, and \( k \) is the upper bound of the size of the skew separator we are looking for.

If \( l = 1 \), then the problem becomes the construction of a min-cut of size bounded by \( k \) from \( S_1 \) to \( T_1 \), which can be solved in \( O(kn^2) \) time by Lemma 2.1. Steps 2-4 were justified in the discussions of Rules 2, 1, 3, respectively, at the beginning of this section (note that we have also consistently defined that an instance is a “No” instance if the parameter \( k \) has a negative value). Therefore, if the algorithm reaches step 5, then none of the Rules 1-3 are applicable. In particular, since Rule 1 is not applicable and the sets \( S_l \) and \( T_{all} \) are disjoint, there must be an edge \([v, w]\), where \( v \in S_l \) and \( w \notin S_l \). Since Rule 2 is not applicable, the vertex \( w \) is not in the set \( T_{all} \). The vertex \( w \) also cannot be in any source set \( S_i \) for \( i < l \) because there is no edge coming into \( S_i \) from outside of \( S_i \). Therefore, the vertex \( w \) is a non-terminal vertex. Finally, since Rule 3 is not applicable, there is no edge from \( w \) to \( T_{all} \). Thus, \( w \) must be an \( S_l \)-extended vertex. This proves that at step 5, the algorithm can always find an \( S_l \)-extended vertex \( u_0 \).

In the case \( m > k \) in step 7, i.e., the size \( m \) of a min-cut from \( S_l \) to \( T_{all} \) is larger than the parameter \( k \), then even separating a single source set \( S_l \) from the sink sets \( T_{all} = \bigcup_{j=1}^{l} T_j \) requires more than \( k \) vertices. Thus, no skew separator of size bounded by \( k \) can exist to separate \( S_l \) from \( T_j \) for all \( l \geq i \geq j \geq 1 \). Step 7 correctly handles this case by returning “No”.

In the case \( m = m' \) in step 9, i.e., the size \( m \) of a min-cut from \( S_l \) to \( T_{all} \) is equal to the size \( m' \) of a min-cut from \( S_l' \) to \( T_{all} \), by Theorem 3.4, the pair \([S_1, \ldots, S_{l-1}, S_l', T_1, \ldots, T_{l-1}, T_l]\) has a skew separator of size bounded by \( k \) if and only if the pair \([S_1, \ldots, S_{l-1}, S_l', T_1, \ldots, T_{l-1}, T_l]\) has a skew separator of size bounded by \( k \). Moreover, by Corollary 3.2, a skew separator of size bounded by \( k \) for the pair \([S_1, \ldots, S_{l-1}, S_l', T_1, \ldots, T_{l-1}, T_l]\) is also a skew separator for the pair \([S_1, \ldots, S_l, T_1, \ldots, T_l]\). Therefore, in this case we can recursively call \( \text{SMC}(G, [S_1, \ldots, S_{l-1}, S_l', T_1, \ldots, T_{l-1}, T_l], k) \), and look instead for a skew separator of size bounded by \( k \) for the pair \([S_1, \ldots, S_{l-1}, S_l', T_1, \ldots, T_{l-1}, T_l]\), as handled by step 9.1.

In the case \( m \neq m' \), then the algorithm branches into two subcases: step 9.2 includes the \( S_l \)-extended vertex \( u_0 \) in the skew separator and recursively looks for a skew separator of size bounded by \( k - 1 \) in the remaining graph \( G - u_0 \) for the pair \([S_1, \ldots, S_l, T_1, \ldots, T_l]\); and step 9.3 excludes the \( S_l \)-extended vertex \( u_0 \) from the skew separator and recursively looks for a skew separator that does not contain \( u_0 \) and is of size bounded by \( k \) in the graph \( G \) for the pair \([S_1, \ldots, S_l, T_1, \ldots, T_l]\) (which, by Lemma 3.1, is a skew separator of size bounded by \( k \) for the pair \([S_1, \ldots, S_{l-1}, S_l', T_1, \ldots, T_{l-1}, T_l]\)). This completes the verification of the correctness of the algorithm. Now we analyze its complexity.

The recursive execution of the algorithm can be described as a search tree \( T \). We first count the number of leaves in the search tree \( T \). Note that only steps 9.2-9.3 of the algorithm correspond to branches in the search tree \( T \). Let \( D(k, m) \) be the total number of leaves in the search tree \( T \) for the algorithm \( \text{SMC}(G, [S_1, \ldots, S_l, T_1, \ldots, T_l], k) \), where \( m \) is the size of a min-cut from \( S_l \) to \( T_{all} \). Then steps 9.2-9.3 induce the following recurrence relation:

\[
D(k, m) \leq D(k - 1, m_1) + D(k, m_2)
\]  

where \( m_1 \) is the size of a min-cut from \( S_l \) to \( T_{all} \) in the graph \( G - u_0 \) as given in step 9.2, and \( m_2 \) is the size of a min-cut from \( S_l' \) to \( T_{all} \) in the graph \( G \) as given in step 9.3. Note that
We prove, by induction on \( t = 2k - m \), that \( D(k, m) \leq 2^{2k-m} \). First note that we always have \( t = 2k - m \geq 0 \) because by the definitions of \( k \) and \( m \) we always have \( k \geq m \geq 0 \). In particular, in the initial case when \( t = 2k - m = 0 \), we must have \( k = m = 0 \); in this case the algorithm can solve the instance without further branching. Therefore, we have \( D(k, m) = 1 \) when \( t = 2k - m = 0 \). For the inductive step, note that by Inequalities (2), we have

\[
t_1 = 2(k-1) - m_1 \leq 2(k-1) - (m-1) = 2k - m - 1,
\]

and

\[
t_2 = 2k - m_2 \leq 2k - (m+1) = 2k - m - 1.
\]

Therefore, we can apply the inductive hypothesis on Inequality (1), which gives

\[
D(k, m) \leq D(k - 1, m_1) + D(k, m_2)
\]

\[
\leq 2^{2(k-1)-m_1} + 2^{2k-m_2}
\]

\[
\leq 2^{2k-m-1} + 2^{2k-m-1} = 2^{2k-m}.
\]

This completes the inductive proof. Moreover, we also note that certain non-branching steps (i.e., steps 3, 4, and 9.1) may also change the values of \( k \) and \( m \), thus changing the value \( t = 2k - m \). However, none of these steps increases the value \( t = 2k - m \): (i) step 3 keeps the value \( k \) unchanged and does not decrease the value \( m \) (because in this case the size of a min-cut from \( S_l \) to \( T_{all} \) is 0 that cannot be larger than the size of a min-cut from \( S_{l-1} \) to \( \bigcup_{j=1}^{l-1} T_j \)); (ii) step 4 decreases the value \( k \) by 1 and the value \( m \) by at most 1 (because removing a vertex from \( G \) can reduce the size of a min-cut from \( S_l \) to \( T_{all} \) by at most 1), which as a total will decrease the value \( t = 2k - m \) by at least 1; (iii) by the condition assumed, step 9.1 keeps both the values \( k \) and \( m \) unchanged, thus unchanging the value \( t = 2k - m \). As a result, the value \( t = 2k - m \) after a branching step to the next branching step can never be increased.

Summarizing the above discussion, we conclude that the total number of leaves, \( D(k, m) \), in the search tree \( T \) for the algorithm \textbf{SMC}(\( G, [S_1, \ldots, S_l, [T_1, \ldots, T_l]], k \)), where \( m \) is the size of a min-cut from \( S_l \) to \( T_{all} \), satisfies the following inequality

\[
D(k, m) \leq 2^{2k-m} \leq 4^k.
\]

The running time of each execution of the algorithm \textbf{SMC}, not counting the time for the recursive calls in the execution, is bounded by \( O(kn^2) \), where \( n \) is the number of vertices in the input graph. In particular, by Lemma 2.1, step 1 that looks for a min-cut of size bounded by \( k \) from \( S_l \) to \( T_1 \), steps 6-7 that determine if the size \( m \) of a min-cut from \( S_l \) to \( T_{all} \) is bounded by \( k \), and steps 8-9 that determine if the size of a min-cut from \( S'_l \) to \( T_{all} \) is equal to \( m \) (\( m \leq k \) at this point), all have their running time bounded by \( O(kn^2) \).
Let $G$ be a directed graph, $(G, \pi)$ is a DAG-bipartition of $G$, and $\pi$ is a skew separator for the pair $(G, \pi)$. In particular, a skew separator for the pair $(G, \pi)$ is a tuple $(\pi, 1, 2, k)$, where $\pi$ is a topologically sorted order of the vertices in the induced DAG $G[\pi]$, every edge going out from a vertex $\pi_i$ must come into a vertex $\pi_{i-1}$, and that the size of $\pi$ is bounded by $k$.

Let $\pi = \{v_1, v_2, \ldots, v_h\}$ be a topologically sorted order of the vertices in the induced DAG $G[D_2]$. We construct an instance of the skew separator problem as follows:

1. Let $G'$ be the graph obtained from $G$ by removing all edges in $G[D_2]$.
2. In the graph $G'$, replace each vertex $v_i$ in $D_2$ by a pair $(t_i, s_i)$ of vertices such that all incoming edges into $v_i$ are now coming into the vertex $t_i$, and that all outgoing edges from $v_i$ are now going out from the vertex $s_i$. Let the resulting graph be $G_\pi$.

Note that in the resulting graph $G_\pi$, the vertices $s_i$, $1 \leq i \leq h$, have no incoming edges, and the vertices $t_j$, $1 \leq j \leq h$, have no outgoing edges. Moreover, since we have removed all edges between the vertices in $G[D_2]$, every edge going out from a vertex $s_i$ must come into a vertex in the set $D_1$, and every edge coming into a vertex $t_j$ must go out from a vertex in the set $D_1$. In particular, $(G_\pi, \{\{s_1\}, \ldots, \{s_h\}, \{t_1\}, \ldots, \{t_h\}, k)$ is a valid instance for the skew separator problem, which will be called an instance of the skew separator induced by the instance $(G, D_1, D_2, k)$ of DAG-bipartition FVS and the topologically sorted order $\pi$ of the vertices in $G[D_2]$.

Thus, each vertex $v_i$ in the set $D_2$ in the graph $G$ is now “split” into the two vertices $s_i$ and $t_i$ in the graph $G_\pi$. Moreover, there is a one-to-one mapping between the vertices in the set $D_1$ in the graph $G$ and the non-terminal vertices in the graph $G_\pi$. Thus, in case of no ambiguity, we will use the same vertex name to refer to both a non-terminal vertex in the graph $G_\pi$ and a vertex in the set $D_1$ in the graph $G$. In particular, a skew separator for the pair $(\{s_1\}, \ldots, \{s_h\}, \{t_1\}, \ldots, \{t_h\})$ in the graph $G_\pi$ corresponds to a subset of $D_1$ in the graph $G$. We have the following important theorem.

### 4 Solving the DAG-BIPARTITION FVS problem

In this section, we describe how to use the results in the previous section to solve the DAG-bipartition FVS problem.

Recall that an instance of DAG-bipartition FVS is given as a tuple $(G, D_1, D_2, k)$, where $G$ is a directed graph, $(D_1, D_2)$ is a DAG-bipartition of $G$, and $k$ is a parameter, with the objective of finding an FVS $X$ for the graph $G$ such that $X \subseteq D_1$ (recall that such an FVS is called a $D_1$-FVS) and that the size of $X$ is bounded by $k$.

Summarizing the above discussions, we conclude that the running time of the algorithm SMC is bounded by $O(4^k kn^3)$. This completes the proof of the theorem. 

Observe that for each recursive call in an execution of the algorithm SMC, either the number of source-sink pairs in the instance is decreased by 1 (step 3), or the number of non-terminal vertices in the instance is decreased by 1 (steps 4, 9.1, 9.2, and 9.3). When the number of source-sink pairs is equal to 1, the problem is solved in time $O(kn^2)$ by step 1, and when the number of non-terminal vertices is equal to 0, either step 2 or step 3 can be applied directly. In conclusion, along each root-leaf path in the search tree $T$, there are at most $O(n)$ recursive calls to the algorithm SMC. Therefore, the running time along each root-leaf path in the search tree $T$ is bounded by $O(kn^3)$.
Theorem 4.1 Let \((G, D_1, D_2, k)\) be an instance of the DAG-bipartition FVS problem, and let \(X\) be a \(D_1\)-FVS for the graph \(G\). Then there is a topologically sorted order \(\pi = \{v_1, \ldots, v_h\}\) of the vertices in \(G[D_2]\) such that in the instance \((G_\pi, \{s_1\}, \ldots, \{s_h\}), \{(t_1), \ldots, (t_h)\}, k\) induced by \((G, D_1, D_2, k)\) and \(\pi\): (1) \(X\) is a skew separator for the pair \((\{s_1\}, \ldots, \{s_h\}), \{(t_1), \ldots, (t_h)\}\) in the graph \(G_\pi\); and (2) every skew separator for the pair \((\{s_1\}, \ldots, \{s_h\}), \{(t_1), \ldots, (t_h)\}\) in \(G_\pi\) is a \(D_1\)-FVS for the graph \(G\).

Proof. As assumed in the theorem, let \((G, D_1, D_2, k)\) be an instance of the DAG-bipartition FVS problem, and let \(X\) be a \(D_1\)-FVS for the graph \(G\). Consider the subgraph \(G - X\). Since \(X\) is an FVS for \(G\), the graph \(G - X\) is a DAG. Therefore, the vertices in \(G - X\) can be topologically sorted into an ordered list \(\pi'\) such that there is no edge in \(G - X\) that goes out from a later vertex in \(\pi'\) and comes into an earlier vertex in \(\pi'\). Let \(\pi = \{v_1, \ldots, v_h\}\) be the order of the vertices in \(D_2\) that is induced from the order \(\pi'\) (i.e., \(\pi\) is obtained from \(\pi'\) by removing the vertices not in \(D_2\). Note that all vertices in \(X\) are in \(D_1\). The order \(\pi\) is obviously a topologically sorted order for the DAG \(G[D_2]\). We show that this order \(\pi\) of the vertices in \(D_2\) and the corresponding instance \((G_\pi, \{s_1\}, \ldots, \{s_h\}), \{(t_1), \ldots, (t_h)\}, k\) induced by \((G, D_1, D_2, k)\) and \(\pi\) satisfy the conclusions of the theorem.

We first show that the set \(X\) is a skew separator for the pair \((\{s_1\}, \ldots, \{s_h\}), \{(t_1), \ldots, (t_h)\}\) in the graph \(G_\pi\). If this were not the case, then there would be a path \(P\) in the graph \(G_\pi - X\) that starts from a vertex \(s_i\) and ends at a vertex \(t_j\) with \(i \geq j\). Since no vertex in \(\{s_1, \ldots, s_h\}\) has incoming edges and no vertex in \(\{t_1, \ldots, t_h\}\) has outgoing edges, all internal vertices on the path \(P\) are non-terminal vertices in \(G_\pi\). In consequence, all internal vertices on \(P\) are vertices in the set \(D_1\) in the graph \(G\). Therefore, the path \(P\) in \(G_\pi - X\) corresponds to a path \(P'\) in the graph \(G - X\) that starts from the vertex \(v_i\) and ends at the vertex \(v_j\), where \(i \geq j\), with all internal vertices of \(P'\) in the set \(D_1\). But it is impossible: (1) if \(i = j\) then the path \(P'\) would be a cycle in the graph \(G - X\), contradicting the assumption that \(X\) is an FVS for the graph \(G\); and (2) if \(i > j\), then \(P'\) would become a path from \(v_i\) to \(v_j\) with \(i > j\) in the graph \(G - X\), contradicting the assumption that \(\pi = \{v_1, \ldots, v_h\}\) is an order of the vertices in \(D_2\) that is induced from the topologically sorted order \(\pi'\) of the vertices in the DAG \(G - X\). In conclusion, the path \(P\) does not exist, and the set \(X\) is a skew separator for the pair \((\{s_1\}, \ldots, \{s_h\}), \{(t_1), \ldots, (t_h)\}\) in the graph \(G_\pi\).

Now we prove that every skew separator \(X'\) for the pair \((\{s_1\}, \ldots, \{s_h\}), \{(t_1), \ldots, (t_h)\}\) in the graph \(G_\pi\) is a \(D_1\)-FVS for the graph \(G\). First of all, by definition, a skew separator consists of only non-terminals, thus, all vertices in \(X'\) are in the set \(D_1\). Suppose for a contradiction that \(X'\) is not a \(D_1\)-FVS for the graph \(G\). Then there is a cycle \(C\) in the graph \(G - X'\). Without loss of generality, we can assume that \(C\) is a simple cycle. Since both the induced subgraphs \(G[D_1]\) and \(G[D_2]\) are DAGs, the cycle \(C\) must contain both vertices in \(D_1\) and vertices in \(D_2\). We consider two different cases.

Case 1. The cycle \(C\) contains a single vertex \(v_i\) in the set \(D_2\). Then all other vertices in the cycle \(C\) are in the set \(D_1\). But then the cycle \(C\) would correspond to a path \(P_1\) in the graph \(G_\pi - X'\) that starts with the vertex \(s_i\) and ends at the vertex \(t_i\) (with all internal vertices being non-terminal vertices). But this contradicts the assumption that \(X'\) is a skew separator for the
The algorithm \( G \) in the input graph \( \mathcal{G} \) time

**Theorem 4.2**

Case 2. The cycle \( C \) contains more than one vertex in the set \( D_2 \). Let \( \{v_1, v_2, \ldots, v_d, v_1\} \) be the order of the vertices in \( D_2 \) that we encounter when traversing along the cycle \( C \) (starting from an arbitrary vertex \( v_i \) in \( D_2 \)), where \( d > 1 \). Then there must be an index \( j \) such that \( i_j > i_{j+1} \) (where we take \( i_{j+1} = i_1 \) if \( j = d \)). Now consider the subpath \( P_2 \) of \( C \) that starts from the vertex \( v_i \) and ends at the vertex \( v_{i_{j+1}} \). The path \( P_2 \) cannot be a single edge from \( v_i \) to \( v_{i_{j+1}} \) since \( \pi = \{v_1, v_2, \ldots, v_h\} \) is a topologically sorted order for the vertices in the DAG \( G[D_2] \) and \( i_j > i_{j+1} \). Thus, the path \( P_2 \) contains at least one internal vertex. Since all internal vertices on the path \( P_2 \) are not in \( D_2 \) thus correspond to non-terminal vertices in the graph \( G_x \) – \( G' \), the path \( P_2 \) would correspond to a path \( P_2' \) in the graph \( G_x \) – \( G' \) that starts from the vertex \( s_i \) and ends at the vertex \( t_{i_{j+1}} \), with \( i_j > i_{j+1} \). Again this contradicts the assumption that \( G' \) is a skew separator for the pair \( ([s_1], \ldots, [s_h]), ([t_1], \ldots, [t_h]) \), which should have cut all paths from \( s_i \) to \( t_{i_{j+1}} \) when \( i_j > i_{j+1} \).

This proves that the skew separator \( X' \) for the pair \( ([s_1], \ldots, [s_h]), ([t_1], \ldots, [t_h]) \) in the graph \( G_x \) must be a \( D_1 \)-FVS for the graph \( G \). This completes the proof of the theorem.

**Algorithm DBF**

For the correctness of the algorithm, first note that the algorithm always correctly reports “No” unless it actually constructs a \( D_1 \)-FVS of size bounded by \( k \) for \( G \) in step 1.3. In particular, if the input instance \((G, D_1, D_2, k)\) contains no \( D_1 \)-FVS of size bounded by \( k \) for the graph \( G \), then the algorithm always correctly reports “No”.

**Theorem 4.2** The algorithm **DBF**\((G, D_1, D_2, k)\) solves the DAG-BIPARTITION FVS problem in time \( O(4^k n h^3 h!) \), where \( h \) is the number of vertices in the set \( D_2 \), and \( n \) is the number of vertices in the input graph \( G \).

**Proof.** The running time of the algorithm is obvious: the for-loop in step 1 is executed at most \( h! \) times, and the time for each execution is dominated by the subroutine call to the algorithm SMC in step 1.2. By Theorem 3.5, the running time of each execution of step 1.2 is bounded by \( O(4^k n^3) \).

For the correctness of the algorithm, first note that the algorithm always returns “No” unless it actually constructs a \( D_1 \)-FVS of size bounded by \( k \) for \( G \) in step 1.3. In particular, if the input instance \((G, D_1, D_2, k)\) contains no \( D_1 \)-FVS of size bounded by \( k \) for the graph \( G \), then the algorithm always correctly reports “No”.

---

**Figure 3:** An algorithm for the DAG-BIPARTITION FVS problem.
On the other hand, suppose that there is a $D_1$-FVS $X_0$ of size bounded by $k$ for the graph $G$. Then by Theorem 4.1, there is a topologically sorted order $\pi = \{v_1, \ldots, v_h\}$ of the vertices in the DAG $G[D_2]$ such that in the instance $(G_\pi, \{(s_1), \ldots, (s_h)\}, \{(t_1), \ldots, (t_h)\}, k)$ of the skew separator problem induced by $(G, D_1, D_2, k)$ and $\pi$, the set $X_0$ is a skew separator for the pair $((s_1), \ldots, (s_h), \{(t_1), \ldots, (t_h)\})$ in the graph $G_\pi$, and every skew separator for the pair $((s_1), \ldots, (s_h), \{(t_1), \ldots, (t_h)\})$ in the graph $G_\pi$ is a $D_1$-FVS for the graph $G$. In particular, the pair $((s_1), \ldots, (s_h), \{(t_1), \ldots, (t_h)\})$ has at least one skew separator of size bounded by $k$ (e.g., $X_0$) in the graph $G_\pi$. Therefore, step 1.2 of the algorithm $DBF$ must return a skew separator $X$ of size bounded by $k$ for the pair $((s_1), \ldots, (s_h), \{(t_1), \ldots, (t_h)\})$ in the graph $G_\pi$ (the set $X$ may be different from the set $X_0$), and this set $X$ is a $D_1$-FVS for the graph $G$. In conclusion, if there is a $D_1$-FVS of size bounded by $k$ for the graph $G$, then the algorithm $DBF(G, D_1, D_2, k)$ will correctly return a $D_1$-FVS of size bounded by $k$ in step 1.3.

\begin{proof}
Let $G = (V, E)$ be the input directed graph with $|V|$ vertices, and let $F$ be the input FVS of size $k + 1$ for the graph $G$. Every FVS $F'$ of size bounded by $k$ for $G$ can be split into two disjoint subsets $F_1$ and $F_2$, where $F_2$ consists of $j$ vertices in $F$ for some integer $j$, $0 \leq j \leq k$, and $F_1$ consists of at most $k - j$ vertices in $V - F$. Note that since we assume that no vertex in $F - F_2$ is in the FVS $F'$, the induced subgraph $G[F - F_2]$ must be a DAG. Therefore, for each $j$, $0 \leq j \leq k$, we enumerate all subsets of $j$ vertices in $F$. For each such subset $F_2$ of $F$ such that $G[F - F_2]$ is a DAG, we seek a subset $F_1$ of at most $k - j$ vertices in $V - F$ such that $F_1 \cup F_2$ makes an FVS for the graph $G$.

Fix a subset $F_2$ of $F$, such that $|F_2| = j$ and that the induced subgraph $G[F - F_2]$ is a DAG. Note that the graph $G$ has an FVS $F_1 \cup F_2$ of size bounded by $k$, where $F_1 \subseteq V - F$, if and only if the subset $F_1$ of $V - F$ is an FVS for the graph $G - F_2$ and the size of $F_1$ is bounded by $k - j$. Therefore, to solve the original problem, we can instead consider how to construct an FVS $F_1$ for the graph $G - F_2$ such that $|F_1| \leq k - j$ and $F_1 \subseteq V - F$.

Since $F$ is an FVS for $G$, we have that the induced subgraph $G[V - F] = G - F$ is a DAG. Moreover, by our assumption, the induced subgraph $G[F - F_2]$ is also a DAG. Note that $(V - F) \cup (F - F_2) = V - F_2$, which is the vertex set for the graph $G' = G - F_2$. Therefore, $(V - F, F - F_2)$ is a DAG-bipartition of the graph $G'$. Thus, an FVS $F_1$ for the graph $G'$ such that $|F_1| \leq k - j$ and $F_1 \subseteq V - F$, is actually a $(V - F)$-FVS of size bounded by $k - j$ for the

\section{Solving the DFVS problem}

We present our algorithm for the DFVS problem. We start with a more restricted version of the problem, the DFVS REDUCTION problem, defined as follows.

**DFVS REDUCTION:** given a triple $(G, F, k)$, where $G$ is a directed graph and $F$ is an FVS of size $k + 1$ for $G$, either construct an FVS of size bounded by $k$ for $G$, or report that no such FVS exists.

**Lemma 5.1** The DFVS REDUCTION problem on a triple $(G, F, k)$ is solvable in time $O(n^3 4^k k^3 k!)$, where $n$ is the number of vertices in the input graph $G$.

\begin{proof}
Let $G = (V, E)$ be the input directed graph with $n = |V|$ vertices, and let $F$ be the input FVS of size $k + 1$ for the graph $G$. Every FVS $F'$ of size bounded by $k$ for $G$ can be split into two disjoint subsets $F_1$ and $F_2$, where $F_2$ consists of $j$ vertices in $F$ for some integer $j$, $0 \leq j \leq k$, and $F_1$ consists of at most $k - j$ vertices in $V - F$. Note that since we assume that no vertex in $F - F_2$ is in the FVS $F'$, the induced subgraph $G[F - F_2]$ must be a DAG. Therefore, for each $j$, $0 \leq j \leq k$, we enumerate all subsets of $j$ vertices in $F$. For each such subset $F_2$ of $F$ such that $G[F - F_2]$ is a DAG, we seek a subset $F_1$ of at most $k - j$ vertices in $V - F$ such that $F_1 \cup F_2$ makes an FVS for the graph $G$.

Fix a subset $F_2$ of $F$, such that $|F_2| = j$ and that the induced subgraph $G[F - F_2]$ is a DAG. Note that the graph $G$ has an FVS $F_1 \cup F_2$ of size bounded by $k$, where $F_1 \subseteq V - F$, if and only if the subset $F_1$ of $V - F$ is an FVS for the graph $G - F_2$ and the size of $F_1$ is bounded by $k - j$. Therefore, to solve the original problem, we can instead consider how to construct an FVS $F_1$ for the graph $G - F_2$ such that $|F_1| \leq k - j$ and $F_1 \subseteq V - F$.

Since $F$ is an FVS for $G$, we have that the induced subgraph $G[V - F] = G - F$ is a DAG. Moreover, by our assumption, the induced subgraph $G[F - F_2]$ is also a DAG. Note that $(V - F) \cup (F - F_2) = V - F_2$, which is the vertex set for the graph $G' = G - F_2$. Therefore, $(V - F, F - F_2)$ is a DAG-bipartition of the graph $G'$. Thus, an FVS $F_1$ for the graph $G'$ such that $|F_1| \leq k - j$ and $F_1 \subseteq V - F$, is actually a $(V - F)$-FVS of size bounded by $k - j$ for the

\end{proof}
graph $G'$ with the DAG-bipartition $(V - F, F - F_2)$. Therefore, the set $F_1$ can be constructed by the algorithm $\text{DBF}(G', V - F, F - F_2, k - j)$.

Since $|F| = k + 1$ and $|F_2| = j$, we have $|F - F_2| = k + 1 - j$. Therefore, the DAG $G[F - F_2]$ contains exactly $k + 1 - j$ vertices. By Theorem 4.2, the running time of the algorithm $\text{DBF}(G', V - F, F - F_2, k - j)$ is bounded by $O(4^{k-j}(k-j)n^3(k+1-j)!)$. Now for all integers $j, 0 \leq j \leq k$, we enumerate all subsets $F_2$ of $j$ vertices in $F$ and apply the algorithm $\text{DBF}(G', V - F, F - F_2, k - j)$ for those $F_2$ such that $G[F - F_2]$ is a DAG. As we discussed above, the graph $G$ has an FVS of size bounded by $k$ if and only if for some $F_2$ of $j$ vertices in $F$, where $0 \leq j \leq k$, the algorithm $\text{DBF}(G', V - F, F - F_2, k - j)$ produces an FVS $F_1$ of size bounded by $k - j$ for the graph $G'$. The running time of this process is bounded by the order of

$$\sum_{j=0}^{k} \binom{k+1}{j} \left(4^{k-j}(k-j)n^3(k+1-j)\right) = O(n^34^k3^kk!)$$

This completes the proof of the lemma.

The rest of our process for solving the original DFVS problem is to apply the iterative compression method. The method was proposed by Reed, Smith, and Vetta [22] and has been used for solving the FEEDBACK VERTEX SET problem on undirected graphs [8, 15, 16]. Here we extend the method and apply it to solve the DFVS problem.

**Theorem 5.2** The DFVS problem is solvable in time $O(n^44^k3^kk!)$. 

**Proof.** Let $(G, k)$ be an instance of the DFVS problem, where $G = (V, E)$ is a directed graph with $n = |V|$ vertices, and $k$ is the parameter. Pick any subset $V_0$ of $k + 1$ vertices in $G$, and let $F_0$ be any subset of $k$ vertices in $V_0$. Note that the set $F_0$ is an FVS of $k$ vertices for the induced subgraph $G_0 = G[V_0]$ since the graph $G_0 - F_0$ consists of a single vertex (note that by our assumption, the graph $G$ contains no self-loops).

Let $V - V_0 = \{v_1, v_2, \ldots, v_{n-k-1}\}$. Let $V_i = V_0 \cup \{v_1, \ldots, v_i\}$, and let $G_i = G[V_i]$ be the subgraph induced by $V_i$, for $i = 0, 1, \ldots, n - k - 1$. Inductively, suppose that for an integer $i$, $0 \leq i < n - k - 1$, we have constructed an FVS $F_i$ of size bounded by $k$ for the induced subgraph $G_i$ (this has been the case for $i = 0$). Without loss of generality, we can assume that the set $F_i$ consists of exactly $k$ vertices – otherwise we simply pick $k - |F_i|$ vertices (arbitrarily) from $G_i - F_i$ and add them to the set $F_i$. Now consider the set $F_{i+1}' = F_i + v_{i+1}$. Since $G_{i+1} - F_{i+1}' = G_i - F_i$ and $F_{i+1}'$ is an FVS for $G_i$, the set $F_{i+1}'$ is an FVS of size $k + 1$ for the induced subgraph $G_{i+1}$. In particular, the triple $(G_{i+1}, F_{i+1}', k)$ is a valid instance for the DFVS REDUCTION problem.

Apply Theorem 5.1 to the instance $(G_{i+1}, F_{i+1}', k)$, which either returns an FVS $F_{i+1}$ of size bounded by $k$ for the graph $G_{i+1}$, or claims that no such FVS exists. It is easy to see that if the induced subgraph $G_{i+1} = G[V_{i+1}]$ does not have an FVS of size bounded by $k$, then the original graph $G$ cannot have an FVS of size bounded by $k$. Therefore, in this case, we can simply stop and conclude that there is no FVS of size bounded by $k$ for the original input graph $G$. On the other hand, suppose that an FVS $F_{i+1}$ of size bounded by $k$ is constructed for the graph $G_{i+1}$ in the above process, then the induction successfully proceeds from $i$ to $i + 1$ with a new pair $(G_{i+1}, F_{i+1})$. 

16
In conclusion, the above process either stops at some point and correctly reports that the input graph \( G \) has no FVS of size bounded by \( k \), or eventually ends with an FVS \( F_{n-k} \) of size bounded by \( k \) for the graph \( G_{n-k} = G[V_{n-k}] = G \).

This process is involved in solving at most \( n-k-1 \) instances \((G_i, F_i, k)\) of the DFVS REDUCTION problem, for \( 0 \leq i \leq n-k-2 \). By Theorem 5.1, the running time of the process is bounded by \( O(n^34^kk!/(n-k-1)) = O(n^44^kk^3k!) \), and the process correctly solves the DFVS problem.

**Remark.** The running time of the algorithm in Theorem 5.2 can be further improved by taking advantage of existing approximation algorithms for the FEEDBACK VERTEX SET problem on directed graphs. Even, Naor, Schieber, and Sudan [12] have developed a polynomial time approximation algorithm for the FEEDBACK VERTEX SET problem that for a given directed graph \( G \), produces an FVS \( F \) of size bounded by \( c \cdot \tau \log \log \tau \log \tau \) in time \( O(n^2M(n)\log^2 n) \), where \( c \) is a constant, \( \tau \) is the size of a minimum FVS for the graph \( G \), and \( M(n) = O(n^{2.376}) \) is the complexity of the multiplication of two \( n \times n \) matrices. Therefore, for a given instance \((G, k)\) of the DFVS problem, we can first apply the approximation algorithm in [12] to construct an FVS \( F \) for the graph \( G \). If \( |F| > c \cdot k \log k \log \log k \), then we know that the graph \( G \) has no FVS of size bounded by \( k \). On the other hand, suppose that \(|F| \leq c \cdot k \log k \log \log k \). Then we pick a subset \( F_0 \) of arbitrary \( k \) vertices in \( F \), and let \( G_0 = G - (F - F_0) \). The set \( F_0 \) is an FVS of size \( k \) for the graph \( G_0 \). Now we can proceed exactly the same way as we did in Theorem 5.2: let \( F - F_0 = \{v_1, v_2, \ldots, v_h\} \), where \( h \leq c \cdot k \log k \log \log k - k \), and let \( V_i = V_0 \cup \{v_1, \ldots, v_i\} \), and \( G_i = G[V_i] \), for \( i = 0, 1, \ldots, h \). By repeatedly applying the algorithm in Lemma 5.1, we can either stop with a certain index \( i \) where the induced subgraph \( G_{i+1} \) has no FVS of size bounded by \( k \) (thus the original input graph \( G \) has no FVS of size bounded by \( k \)), or eventually construct an FVS \( F_h \) of size bounded by \( k \) for the graph \( G_h = G[V_h] = G \). This process calls for the execution of the algorithm in Lemma 5.1 at most \( h = O(k \log k \log \log k) \) times, and each execution takes time \( O(n^34^kk^3k!) \). In conclusion, the DFVS problem can be solved in time \( O(n^34^kk^4k! \log k \log \log k + n^{4.376} \log^2 n) \), where the second term in the complexity is due to the approximation algorithm given in [12].

**6 Remarks and future research**

We presented a parameterized algorithm of running time \( O(n^44^kk^3k!) \) for the DFVS problem, which shows that the problem is fixed-parameter tractable, and resolves an outstanding open problem in parameterized computation and complexity. Before we close the paper, we give a few remarks on our results and on directions for future research.

There is an edge version of the FEEDBACK SET problem, which is called the FEEDBACK ARC SET problem (briefly, the DFAS problem): given a directed graph \( G \) and a parameter \( k \), either construct a set of at most \( k \) edges in \( G \) whose removal leaves a DAG, or report that no such edge set exists. The DFAS problem is also a well-known NP-complete problem [14]. As shown by Even, Naor, Schieber, and Sudan [12], the DFAS problem and the DFVS problem can be reduced in linear time from one to the other with the same parameter. Therefore, our results also imply an \( O(n^44^kk^3k!) \) time algorithm for the DFAS problem.
The techniques developed in this paper for solving the skew separator problem seem to be powerful and generally useful in the study of a variety of separator problems. For example, it has been used recently in developing improved algorithms for a multi-cut problem on undirected graphs in which a separator is sought to (uniformly) separate a set of given terminals [7]. It will be interesting to identify the conditions for the multi-cut problems under which these techniques (and their variations and generalizations) are applicable. In particular, it will be interesting to see if the techniques are applicable to derive the fixed-parameter tractability of the feedback vertex set problem on weighted and directed graphs. Note that the fixed-parameter tractability of the problem on weighted and undirected graphs has been derived recently [5].

It will be interesting to develop new techniques that lead to faster parameterized algorithms for the dfvs problem and other related problems. For example, is it possible that the dfvs problem can be solved in time $O(c^k n^{O(1)})$ for a constant $c$? Another direction is to look at the kernelization of the dfvs problem, by which we refer to a polynomial-time algorithm that on an instance $(G, k)$ of the dfvs problem, produces a (smaller) instance $(G', k')$ of the problem, such that the size of the graph $G'$ (the kernel) is bounded by a function $g(k)$ of $k$ (but independent of the size of the original graph $G$), that $k' \leq k$, and that the graph $G$ has an FVS of size bounded by $k$ if and only if the graph $G'$ has an FVS of size bounded by $k'$. Since now it is known that the dfvs problem is fixed-parameter tractable, by a general theorem in parameterized complexity theory [11], such a kernelization algorithm exists for the dfvs problem. However, how small can the size of the kernel $G'$ be? In particular, can the kernel $G'$ have its size bounded by a polynomial of the parameter $k$? We note that recently there has been progress in the study of kernelization for the feedback vertex set problem on undirected graphs. Bodlaender [3] was able to give a kernel of size $O(k^3)$ for the feedback vertex set problem on undirected graphs, and Bodlaender and Penninkx [4] have shown that the feedback vertex set problem on undirected planar graphs has a kernel of size $O(k)$.

Acknowledgements. J. Chen, Y. Liu, and S. Lu were supported in part by the US National Science Foundation under the Grant CCF-0430683. B. O’Sullivan and I. Razgon were supported by Science Foundation Ireland Grant 05/IN/I886. The authors are grateful to Hans Bodlaender, Daniel Marx, Mike Fellows, Mike Langston, Vijaya Ramachandran, and Venkatesh Raman for their useful discussions and comments. Daniel Marx suggested the name “skew separator”, and Hans Bodlaender suggested the further improvement on the algorithm’s running time via approximation algorithms. The authors would like to thank an anonymous referee who pointed out a technical bug, which has been fixed, and suggested a possible correction.

References


